Pengantar Pemrograman C#

[Lecture Note Pertemuan ke - 07]
(Penanganan Kesalahan)

Ng Poi Wong, 2019, Sesi 7 : Penanganan Kesalahan, Lecture Notes, Pengantar Pemrograman C# (IF0054), STMIK Mikroskil Medan, Dikirimkan 02 September 2019.

Capaian MK : Mahasiswa mampu menerapkan penanganan kesalahan

DAFTAR ISI PENJELASAN SLIDE

PENJELASAN DARI SLIDE ke-3 s/d 9

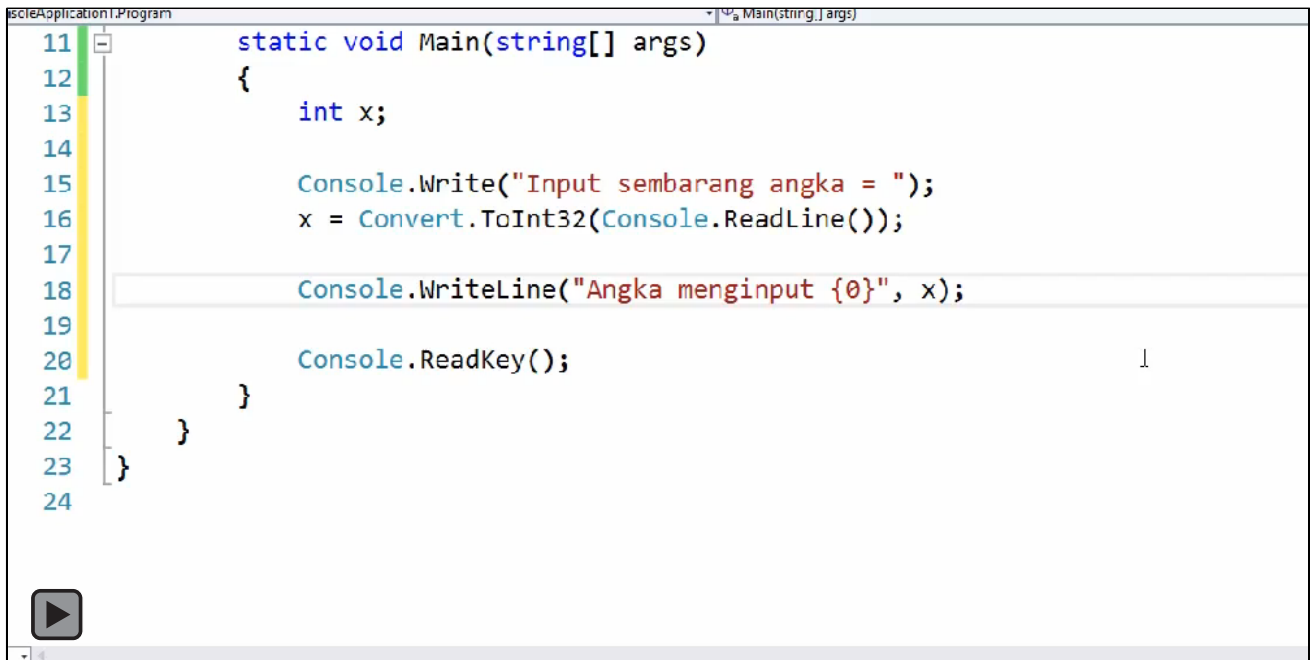
[Slide 3]
Merupakan struktur penggunaan penanganan kesalahan (Exception), dimana terdiri dari 3 blok :

- Try : Selalu dijalankan/dieksekusi.
- Catch : Dijalankan apabila dari blok Try terdapat kesalahan (Exception).
- Finally : Selalu dijalankan/dieksekusi setelah blok Try atau blok Catch selesai dijalankan.

[Slide 4 & 5]
Dari contoh kode program tersebut, apabila diinput dengan nilai yang mengandung karakter (String), maka akan terjadi Error dengan pesan "**FormatException was unhandled, Input string was not in a correct format**"

[Slide 6 & 7]
Dari contoh kode program tersebut, dapat dijelaskan :

```
int x = 0;
Console.Write("Input sembarang angka = ");
try {
  x = Convert.ToInt32(Console.ReadLine());  → Selalu dijalankan.
}
catch (FormatException e) {
  x = -1;  → Akan dijalankan apabila terjadi kesalahan format input nilai pada blok Try di atas.
}
finally {
  Console.WriteLine("Angka menginput {0}", x); → Selalu dijalankan setelah blok Try atau
                                                  blok Catch dijalankan.
}
```

[Slide 8 & 9]

- Dari contoh kode program dari **[Slide 6 & 7]**, dapat dihilangkan blok Finally, karena blok Finally bersifat Opsional, dimana terjadinya Exception atau tidak, blok Finally tetap dijalankan.
- Penulisan parameter pada blok Catch adalah bersifat Opsional. Apabila tidak ada parameter pada blok Catch, maka Exception yang akan ditangkap adalah semua Exception yang bakal terjadi, tanpa melihat jenis Exception.
- Contoh :

```
int x = 0;
Console.Write("Input sembarang angka = ");
try {
  x = Convert.ToInt32(Console.ReadLine());  → Selalu dijalankan.
}
catch {
  x = -1;  → Akan dijalankan apabila terjadi kesalahan (apapun jenis kesalahannya) pada blok
             Try di atas.
}
finally {
  Console.WriteLine("Angka menginput {0}", x); → Selalu dijalankan setelah blok Try
                                                  atau blok Catch dijalankan.
}
```

PENJELASAN DARI SLIDE ke-3 s/d 9

[Video Ilustrasi]
Note : Memerlukan Adobe Flash Player & Video memiliki Audio.

```
11          static void Main(string[] args)
12          {
13              int x;
14
15              Console.Write("Input sembarang angka = ");
16              x = Convert.ToInt32(Console.ReadLine());
17
18              Console.WriteLine("Angka menginput {0}", x);
19
20              Console.ReadKey();
21          }
22      }
23  }
24
```

PENJELASAN DARI SLIDE ke-10 s/d 16

[Slide 10]
Terdapat 2 jenis Exception di dalam Visual Studio, yakni :

- **Exception Built-In** : Jenis Exception yang sudah tersedia di dalam Visual Studio.
- **Exception Buatan** : Jenis Exception buatan manual untuk kasus-kasus tertentu.

[Slide 11 & 12]
Merupakan daftar Exception Built-In yang umum terjadi dan digunakan.

[Slide 13 s/d 16]
Dari contoh kode program tersebut, dapat dijelaskan :

```
int a, b, c;
try {
  Console.Write("A = ");
  a = Convert.ToInt32(Console.ReadLine());
  Console.Write("B = ");
  b = Convert.ToInt32(Console.ReadLine());
  c = a * b;
  Console.WriteLine("C = {0}", c);
  Console.WriteLine("D = {0}", a / b);
}
```

→ Dari blok Try ini, akan terdapat 3 kemungkinan Exception yang bisa terjadi saat user menginput nilai, yakni :

1. Nilai yang diinput ke dalam variabel **a**, **b**, atau **c** mengandung karakter (String).
2. Nilai yang diinput ke dalam variabel **b** adalah bernilai 0 (Akan terjadi pembagian dengan 0).
3. Nilai yang diinput ke dalam variabel **a**, **b**, atau **c** melebihi kapasitas dari tipe data **Int**.

```
catch (FormatException e) {
  Console.WriteLine("\nTerjadi Exception :\n\"{0}\"", e.Message);
}
```

→ Dijalankan apabila nilai yang diinput ke dalam variabel **a**, **b**, atau **c** mengandung karakter (String).

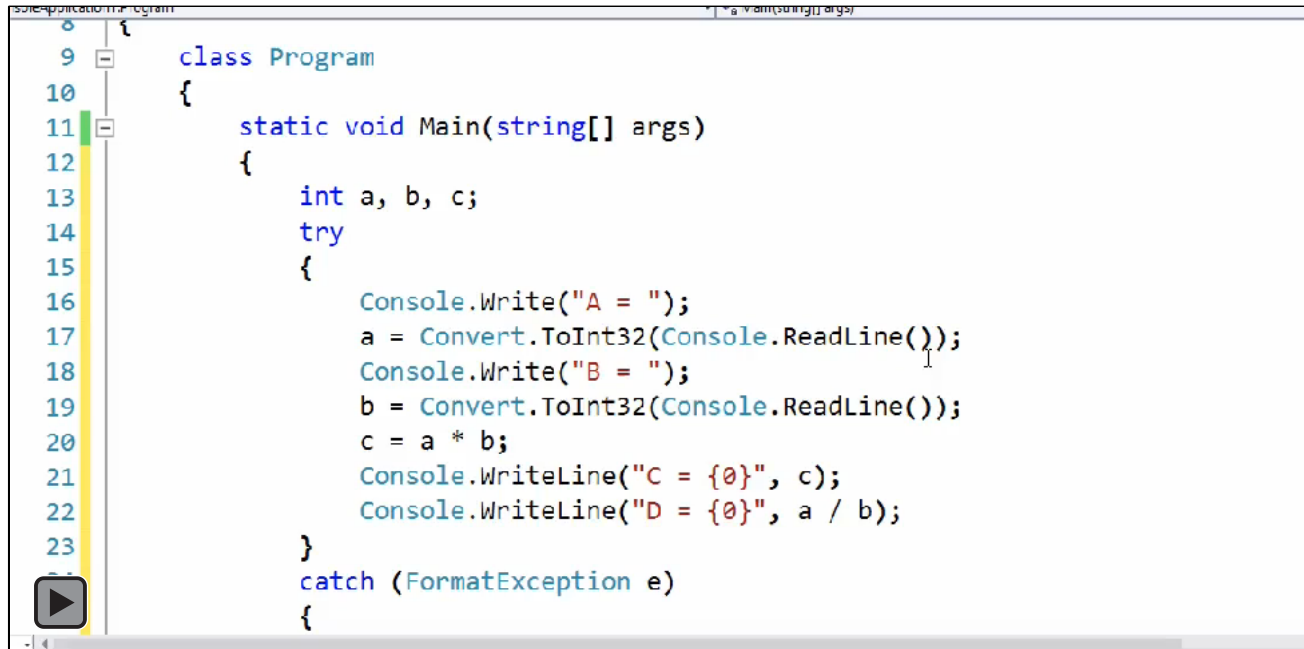
```
catch (DivideByZeroException e) {
  Console.WriteLine("\nTerjadi Exception :\n\"{0}\"", e.Message);
}
```

→ Nilai yang diinput ke dalam variabel **b** adalah bernilai 0 (Akan terjadi pembagian dengan 0)

```
catch (OverflowException e) {
  Console.WriteLine("\nTerjadi Exception :\n\"{0}\"", e.Message);
}
```

→ Nilai yang diinput ke dalam variabel **a**, **b**, atau **c** melebihi kapasitas dari tipe data **Int**.

PENJELASAN DARI SLIDE ke-17 s/d 22

Dari contoh kode program tersebut, dapat dijelaskan :

```
int a, b, c;
try {
  Console.Write("A = ");
  a = Convert.ToInt32(Console.ReadLine());
  Console.Write("B = ");
  b = Convert.ToInt32(Console.ReadLine());
  if ((a < -99999) || (a > 99999))
    throw new Exception("Nilai A antara -99999 s/d 99999");
  else if (b == 0)
    throw new Exception("Nilai B tidak boleh 0");
  else if ((b < -99999) || (b > 99999))
    throw new Exception("Nilai B antara -99999 s/d 99999");
  c = a * b;
  Console.WriteLine("C = {0}", c);
  Console.WriteLine("D = {0}", a / b);
}
catch (Exception e) {
  Console.WriteLine("\nTerjadi Exception :\n\"{0}\"", e.Message);
}
```

→ Akan dijalankan apabila terjadi Exception dari blok Try di atas berupa :

1. Jika Nilai yang diinput ke dalam variabel **a** di luar dari -99999 s/d 99999.
2. Jika Nilai yang diinput ke dalam variabel **b** adalah bernilai 0.
3. Jika Nilai yang diinput ke dalam variabel **c** di luar dari -99999 s/d 99999.

PENJELASAN DARI SLIDE ke-17 s/d 22

[Video Ilustrasi]
Note : Memerlukan Adobe Flash Player & Video memiliki Audio.

```
 9      class Program
10      {
11          static void Main(string[] args)
12          {
13              int a, b, c;
14              try
15              {
16                  Console.Write("A = ");
17                  a = Convert.ToInt32(Console.ReadLine());
18                  Console.Write("B = ");
19                  b = Convert.ToInt32(Console.ReadLine());
20                  c = a * b;
21                  Console.WriteLine("C = {0}", c);
22                  Console.WriteLine("D = {0}", a / b);
23              }
                catch (FormatException e)
                {
```